MS-DOS 2.1 Reference
For the PlusPC

Rev. A

COPYRIGHT

©1985 by VICTOR®.
©1985 by Microsoft® Corporation.

Published by arrangement with Microsoft Corporation, whose software has been customized for use on various desktop microcomputers produced by VICTOR. Portions of the text hereof have been modified accordingly.

All rights reserved. This manual contains proprietary information which is protected by copyright. No part of this manual may be reproduced, transcribed, stored in a retrieval system, translated into any language or computer language, or transmitted in any form whatsoever without the prior written consent of the publisher. For information contact:

VICTOR Publications
380 El Pueblo Road
Scotts Valley, California 95066
(408) 438-6680

TRADEMARKS

VICTOR is a registered trademark of Victor Technologies, Inc.
MS- is a trademark of Microsoft Corporation.
Microsoft is a registered trademark of Microsoft Corporation.
IBM is a trademark of International Business Machines Corporation.

NOTICE

VICTOR makes no representations or warranties of any kind whatsoever with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. VICTOR shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

VICTOR reserves the right to revise this publication from time to time and to make changes in the content hereof without obligation to notify any person of such revision or changes.

Note: The following MS-DOS 2.1 utilities have been copyrighted by MetaWare Inc.: search, fgrep, concat, tail, wc, ls, mv.

First VICTOR printing June, 1985.

ISBN 0-88182-143-8 Printed in U.S.A.

Important Software Diskette Information

For your own protection, do not use this product until you have made a backup copy of your software diskette(s). The backup procedure is described in the user's guide for your computer.

Please read the DISKID file on your new software diskette. DISKID contains important information including:

- The part number of the diskette assembly.
- The software library disk number (for internal use only).
- The date of the DISKID file.
- A list of files on the diskette, with version number, date, and description for each one.
- Configuration information (when applicable).
- Notes giving special instructions for using the product.
- Information not contained in the current manual, including updates, any known bugs, additions, and deletions.

To read the DISKID file onscreen, follow these steps:

1. Load the operating system.
2. Remove your system diskette and insert your new software diskette.
3. Enter —

   type diskid(cr)

4. The contents of the DISKID file is displayed on the screen. If the file is large (more than 24 lines), the screen display will scroll. Type CTRL-S to freeze the screen display; type CTRL-S again to continue scrolling.

Preface

This manual contains these seven chapters and four appendixes:

- Chapter 1 discusses the structure of MS-DOS, how the system uses memory and disk space, how to load MS-DOS, and other introductory topics.
- Chapter 2 describes the types of disk files: system files and utilities, data/text files, and batch files. This chapter also describes how to use device files, and gives the rules for naming files and for using wildcard characters.
- Chapter 3 discusses the hierarchical (tree-structured) MS-DOS directory system, and tells how to use subdirectory names, create and delete directories, move between directories, and find files in the directory system.
- Chapter 4 explains how to use EDLIN, a text editor used mainly to edit source files.
- Chapter 5 tells how to edit MS-DOS command lines, how to redirect input/output, and how to "filter" and "pipe" data.
- Chapter 6 describes all the MS-DOS commands—the internal, external, and batch commands, and the system configuration commands. This chapter gives the syntax for each command, explains parameters used with each command, and gives examples of usage.
- Chapter 7 lists and describes messages the system might display during bootup and during normal operations.
- Appendix A is an ASCII to hexadecimal conversion table.
- Appendix B lists the device input/output errors.
- Appendix C describes the ANSI escape sequences for IBM mode.
- Appendix D contains information on the DEBUG program. All the DEBUG commands are listed alphabetically with a description.

Manual Conventions

This manual uses the following conventions:

- When commands are explained for the first time, the command format is set off in boldface, such as:

  DO filename /switch(cr)

  —The parts of the command that you type exactly as shown (such as the command name) are uppercase.

  —Parts of the command that you vary or omit to fit the situation (such as filenames and drive names) are lowercase.

  —When you type a command, you can use upper- or lowercase for any command elements.

- In examples and command formats, pressing the Return key is shown as (cr), and typing a space is shown as (sp).

- Some examples show the screen, including what you type and the system's response. In these screen examples, what you type is underlined and lowercase. For example:

```
A>print myfile(cr)
```

- In the text, names of commands, files, and programs appear in uppercase. Examples are COPY and TEXT.DOC.

- The CTRL key acts as the "Control" key on your computer. In text, "CTRL-" represents the CTRL key. For example, CTRL-C represents typing the control function of key C. In screen examples, however, the CTRL key is shown as a caret (^), such as ^C.

Contents

Preface V

Manual Conventions VII

1. Introduction
   - 1.1 MS-DOS System Structure 1-1
     - 1.1.1 The PlusPC System Diskette 1-5
     - 1.1.2 Memory 1-6
     - 1.1.3 Disk Space 1-7
   - 1.2 System Configuration 1-8
   - 1.3 Loading MS-DOS 1-8

2. The MS-DOS File System
   - 2.1 Types of Files 2-1
   - 2.2 Filename Conventions 2-2
     - 2.2.1 Filenames 2-3
     - 2.2.2 Filename Extensions 2-3
     - 2.2.3 Drive Names 2-5
   - 2.3 Filename Wild-Card Characters 2-6
   - 2.4 Device Names 2-8
   - 2.5 Batch Files 2-10
     - 2.5.1 Batch File Conventions 2-11
     - 2.5.2 Replaceable Parameters 2-11
     - 2.5.3 The AUTOEXEC.BAT File 2-13

3. The MS-DOS Subdirectory System
   - 3.1 Using Directory Names (Paths) to Locate Files 3-3
     - 3.1.1 Examples of Paths 3-5
     - 3.1.2 Working with Files in Subdirectories 3-6
     - 3.1.3 Paths and External Commands 3-8
   - 3.2 Listing a Directory 3-8

3.3 Making a Subdirectory 3-10
3.4 Moving into a Directory 3-15
3.5 Removing a Subdirectory 3-17

4. Using the Line Editor (EDLIN)
4.1 Invoking EDLIN 4-1
4.1.1 Creating a File 4-2
4.1.2 Editing an Existing File 4-4
4.2 Using File Commands 4-5
4.2.1 File Command Parameters 4-6
4.2.2 File Command Conventions 4-7
4.2.3 Descriptions of File Commands 4-8

5. MS-DOS Command Editing, Filtering, and Piping
5.1 Special MS-DOS Editing Keys 5-1
5.1.1 Command-Line Editing in I Mode 5-3
5.1.2 Command-Line Editing in V Mode 5-6
5.2 Filtering Data 5-10
5.3 Piping Data 5-12
5.3.1 Filtering and Piping Data Files 5-13
5.3.2 Filtering and Piping Columnar Data 5-14
5.3.3 Filtering and Piping Batch Files 5-14

6. MS-DOS Commands
6.1 Overview of Commands 6-1
6.1.1 Internal and External Commands 6-3
6.1.2 Batch Commands 6-6
6.2 Command Syntax and Notation 6-6
6.3 System Configuration Commands 6-9
6.3.1 Descriptions of Configuration Commands 6-10
6.3.2 Sample Configuration File 6-16
6.4 Command Descriptions 6-16

7. MS-DOS Messages 7-1

APPENDIXES

A: ASCII Character Codes A-1
B: Device I/O Errors B-1
C: ANSI Escape Sequences C-1
D: DEBUG D-1

FIGURES

1-1: MS-DOS and Its Resources 1-4
5-1: The Command Line and the Template 5-2
6-1: MS-DOS Command Interface 6-4

TABLES

2-1: MS-DOS File Extensions 2-4
2-2: MS-DOS Device Names 2-9
4-1: EDLIN File Commands and Parameters 4-6
5-1: The I Mode Editing Functions 5-3
5-2: The V Mode Editing Functions 5-7
6-1: Matching Constraint Switches for SEARCH 6-105
6-2: Action Switches for SEARCH 6-110
6-3: SEARCH Action and Constraint Modifiers 6-112
C-1: Set Graphics Rendition Escape Sequence Parameters C-7
C-2: Set Mode Escape Sequence Parameters C-8
D-1: Methods to Start DEBUG D-1
D-2: DEBUG Commands D-4
D-3: Command Parameters D-5
D-4: Register Flag Codes D-22

Chapters

1. Introduction 1

2. The MS-DOS File System 2

3. The MS-DOS Subdirectory System 3

4. Using the Line Editor (EDLIN) 4

5. MS-DOS Command Editing, Filtering, and Piping 5

6. MS-DOS Commands 6

7. MS-DOS Messages 7

Appendixes

A. ASCII Character Codes ..... A

B. Device I/O Errors ..... B

C. ANSI Escape Sequences C

D. DEBUG ..... D

1

Introduction

MS-DOS (Disk Operating System) is a group of programs that:

- Manage files and directories
- Process commands
- Control application programs designed for MS-DOS
- Control the operation of your keyboard, monitor, disk drive(s), and printer(s)

The first part of this chapter gives short descriptions of the system files and utilities that make up MS-DOS. There is a brief explanation of how the PlusPC stores the system and program files on the system diskette. Also included is a discussion of the way MS-DOS uses memory (RAM) and disk space. The last part of the chapter describes system configuration and the procedures for loading MS-DOS and for entering the date and time.

1.1 MS-DOS System Structure

The PlusPC contains three system files/programs. There are two versions of each file: one for I mode, and one for V mode. Each version is prefaced with an "I" for IBM, or a "V" for VICTOR. MS-DOS consists of the following files/programs:

- System Files

  ICOMMAND.COM/VCOMMAND.COM—the command processor that accepts commands from the keyboard and runs the programs that process these commands.

  ICONFIG.SYS/VCONFIG.SYS—system initialization routines and commands for changing the system configuration (see Chapter 6.3).

IDOS.SYS/VDOS.SYS—the operating system programs that manage files and application programs. These system files are "hidden"; that is, they do not appear in directory listings.

▶ PlusPC External Commands

1

ASSIGN assigns a logical designator to a drive.

CHKDSK displays the amount of disk space occupied by files and tells you how much space is available for storage. CHKDSK also scans the directory for errors and can correct allocation problems.

COMMAND loads the command processor (ICOMMAND.COM/VCOMMAND.COM) into memory.

COMP compares the contents of one file to the contents of another file.

CONCAT concatenates (combines) files, displays the files onscreen, or sends them to a file.

DISKCOMP compares the files of two diskettes.

DISKCOPY copies the contents of one diskette onto another diskette. DISKCOPY works in V mode only.

EDLIN is the MS-DOS line editor program.

EXE2BIN converts executable files (.EXE) to the binary format (.COM).

EXESIZE sets a specified file's .EXE header values.

FGREP searches through files or directories for a specified expression, and then prints the names of the files and the lines containing the expression.

FILCOM (file comparison utility) compares two files and displays the differences between the two.

FIND searches for a specified string of text.

FORMAT prepares a diskette to receive MS-DOS files. FORMAT formats diskettes in VICTOR format and works in V mode only.

GRAPHICS sets up a graphics printer to allow printing of graphics display screens. You can use GRAPHICS in I mode only.

LS alphabetically lists a directory, or lists selected parts of a directory according to specified constraints.

MODCON changes the "soft console" (keyboard and character set files) used by your computer, in V mode only.

MODE sets up printer ports and asynchronous communications adapters. There is one MODE command, but some of the forms of this command do not pertain to V mode.

MORE displays the contents of files one screen at a time.

MV moves files (makes a copy of a file, and deletes the original), and renames files.

PRINT queues a list of files for printing.

RECOVER recovers lost data from a file or from a whole disk.

RETROSYS copies the system files to VICTOR format diskettes or to the fixed disk.

SDCOPY copies the contents of one diskette onto another diskette. SDCOPY requires only one floppy drive. You can use SDCOPY in V mode only, and you can use only VICTOR format diskettes.

SEARCH selects files from a given set of files according to matching constraints, and performs a specified action on the matching files.

SORT sorts specific lines of data from a file or arranges the file in ascending or descending order.

TAIL displays the final lines of a file.

TREE lists all directory paths on a drive.

WC counts words and/or lines in a file or a group of files.

Note: Chapter 6.4 contains detailed descriptions of all the MS-DOS commands alphabetically. Refer to Chapter 6.4 for explanations of commands as you read the earlier chapters.

The hidden system file IDOS.SYS/VDOS.SYS contains programs that execute automatically when MS-DOS loads into memory. IDOS.SYS/VDOS.SYS and ICOMMAND.COM/VCOMMAND.COM control the system resources. The relationship between these files and system resources is shown in Figure 1-1. The relationship of the two files remains the same in either mode.

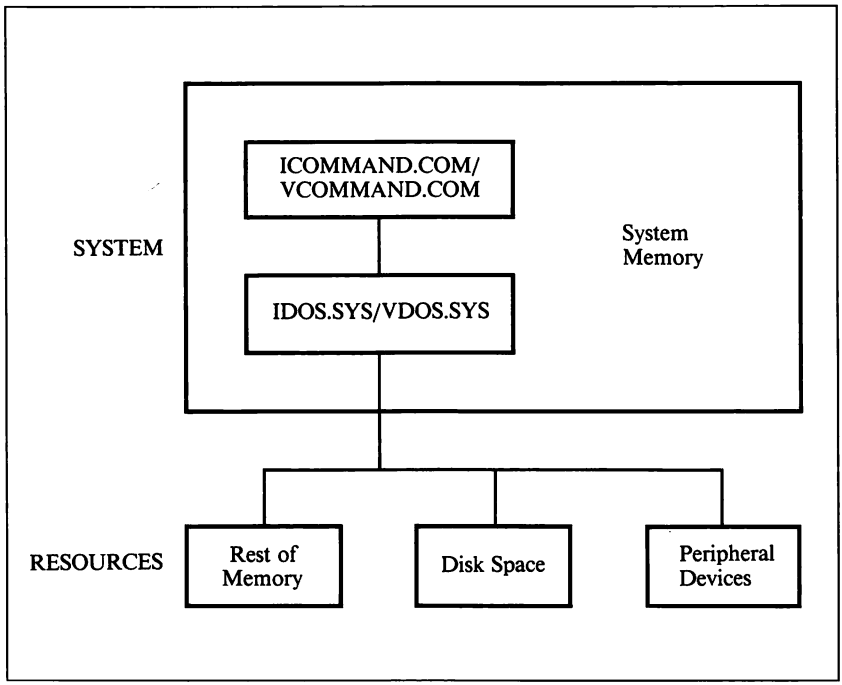


Figure 1-1: MS-DOS and Its Resources

The system's resources are its internal memory (RAM) and external storage (disk space), plus connections to the CRT, keyboard, printer(s), other peripheral devices, and a network configuration, if any.

Memory and the MS-DOS file system can be thought of as the external organization of system resources. MS-DOS supports "device-independent I/O." This means that you can treat devices (such as your printer, keyboard, and CRT) as if they are files when you do input/output processing (see Chapter 2.2). The system's memory and disk space are described in Chapters 1.1.2 and 1.1.3.

1.1.1 The PlusPC System Diskette

On your PlusPC system diskette, VICTOR has provided a subdirectory structure for storing the MS-DOS system and program files. The program files for the PlusPC MS-DOS commands are contained in three different directories called libraries:

- VLIB is the V mode library. The V mode library contains the program files that are specific to the V mode of your PlusPC system.
- ILIB is the I mode library. The I mode library contains the program files that are specific to the I mode of your PlusPC system.
- COMNLIB is the common library. The common library contains the program files that are common, or shared by both the V and I modes of your PlusPC system.

See the *PlusPC User's Guide* for a list of the program files contained in each library. In Chapter 6.4 of this manual, all the MS-DOS commands have special notes if any of their features or parameters are specific to either I mode or V mode.

CAUTION: It is important that you maintain the subdirectory structure provided for you on the system diskette. A path has been set up on the diskette so that the correct version of a program is executed for the mode you are using. If you move any of the MS-DOS program files from the subdirectories in which they are located when you receive the system diskette, you might use the incorrect program. If this happens you can overwrite or reformat existing data. By maintaining the MS-DOS subdirectory structure as it is already set up, you are ensured of using the correct version of each command in the proper mode.

1.1.2 Memory

MS-DOS controls main memory (RAM) as well as disk space and other devices. MS-DOS can load files into memory as data files or as executable files. The actual loading of files is performed by either IDOS.SYS or VDOS.SYS, the lowest level of the MS-DOS operating system. ICOMMAND.COM or VCOMMAND.COM supervises the loading of executable files depending in which mode you are.

Most programs return control to MS-DOS after normal or unexpected termination of the program.

MS-DOS can also overlay (write over) the transient portion of ICOMMAND.COM or VCOMMAND.COM to make room for a particularly large executable file. After such a file is executed, MS-DOS automatically loads the overlaid part of ICOMMAND.COM or VCOMMAND.COM back into memory from disk, and normal execution of ICOMMAND.COM or VCOMMAND.COM resumes. MS-DOS attempts to reload the transient portion from the root directory of the default drive at startup (not from the current default drive, if the default drive has changed) or from the path specified in COMSPEC in the ICONFIG.SYS/VCONFIG.SYS file.

If MS-DOS finds an incorrect version of ICOMMAND.COM or VCOMMAND.COM in attempting to load the overlaid section, it displays a message such as:

```
Invalid ICOMMAND.COM
  COMMAND.COM expected at X:\ICOMMAND.COM
  <CR> for retry, or new COMMAND.COM path =
```

where drive X is the original default drive, and VCOMMAND.COM can be substituted for ICOMMAND.COM depending on the current mode.

Insert the system diskette you used to load MS-DOS (or any diskette with the version of ICOMMAND.COM or VCOMMAND.COM that was originally loaded). To use another drive or path for reloading ICOMMAND.COM or VCOMMAND.COM, type the name of that drive or path name. Now MS-DOS will use that drive or path each time it reloads ICOMMAND.COM or VCOMMAND.COM during this operating session.

1.1.3 Disk Space

In MS-DOS, disk space is divided into four parts:

- The reserved sector contains boot information used each time MS-DOS is loaded.
- The directory contains information about each file on a given diskette. This information includes the file's complete filename, its size, and the time and date of its last modification.
- The file allocation table (FAT) contains location information for the data making up each file on a diskette.
- Files occupy the majority of disk space. An individual file does not necessarily reside in contiguous areas on the diskette; its contents might be "scattered" on the diskette so that disk space is not wasted.

1.2 System Configuration

With MS-DOS your system configuration can include the following peripheral devices:

- One parallel printer
- Two serial devices (a combination of MODEMs, printers, and so on)

To set up your operating system for your particular configuration, refer to the *PlusPC User's Guide.* Chapter 6 in this manual discusses the system configuration commands and the CAUX, CLST, and CTTY commands.

1.3 Loading MS-DOS

Before you start working with your system, you should have a backup copy of your MS-DOS system diskette. Refer to the *PlusPC User's Guide* for instructions on how to make backup copies of the diskette. To load MS-DOS, insert the system diskette in the drive as described in the *PlusPC User's Guide.*

Once MS-DOS is in main memory, it prompts you to enter the date:

```
Enter new date:_
```

MS-DOS keeps track of the date and time. If you enter the date and time when you load MS-DOS, it records the current date and time on each file that you create or change.

To set the date, follow these guidelines:

- Type the date using numbers separated by slashes or by hyphens, in one of these formats:

  —month/day/year or month-day-year (for American OS)

  —day-month-year or day/month/year (for European OS)

- For the month, enter a number from 1 to 12. For the day, enter a number from 1 to 31. For the year, enter a number from 1980 to 2099. Then press Return to submit your entry to MS-DOS.
- Use the Backspace key to correct mistakes.
- Do not type the name of the day because MS-DOS computes the day automatically.
- If you do not want to type in a date, press the Return key. Then MS-DOS uses the system creation date or the last valid date setting as the current date.

To enter the date of February 25, 1985, for example, you would type:

2/25/85 or 2-25-85 (American OS)

or

25-2-85 or 25/2/85 (European OS)

If the date you enter is not in the correct format or contains a number outside the valid ranges (such as 13 for the month), MS-DOS displays:

```
Invalid date:
Enter new date:_
```

After MS-DOS accepts your entry for the date, it displays its current time setting and prompts you to reset the time:

```
Enter new time:_
```

1

Note: The SHELL command in the CONFIG.SYS file can instruct MS-DOS to load the command processor without prompting you for the date and time at bootup.

To enter the time, use a 24-hour clock. For example, 1:00 pm is 13:00, and 11:00 pm is 23:00. When MS-DOS displays the time, it gives the hour, minute, second, tenths and hundredths of a second (0:00:00.00).

To change the time:

- Use the numbers 0–23 for the hour and 0–59 for the minutes. You do not have to make an entry for minutes or seconds. Do not type the tenths and hundredths of a second.
- Separate the time with colons (hour:minute:second) or with periods (hour.minute.second).
- Use the Backspace key to correct mistakes.
- Press the Return key when you want MS-DOS to accept your entry.

If you make an error, after you press the Return key MS-DOS displays:

```
Invalid time:
Enter new time:_
```

Retype the time.

After MS-DOS accepts your entry for the time, the MS-DOS command prompt and the cursor appear on your screen. By default, the prompt displays the name of the drive from which MS-DOS was loaded. For example, if MS-DOS is loaded from drive A, you will see one of the following prompts, depending on whether you are in I or V mode:


```
{ Mode I } A:\>_
```

or

```
Mode V A:\>_
```

Note that "Mode V" in the V mode prompt appears in reverse video.

Because the MS-DOS hierarchical directory structure functions the same for either mode, this manual refers to drive A as A>, drive B as B>, and so on.

The command prompt indicates that MS-DOS is in main memory and is ready for you to enter an MS-DOS command from the keyboard. You can change the MS-DOS prompt to be any letter(s), word, or phrase by using the PROMPT command (see Chapter 6.4 for more information on PROMPT).

You can check and change the date or time while you are using MS-DOS. Type the word **date** or **time** and press Return. MS-DOS shows the date or time and asks you to reset it. Press Return to leave it as is, or follow the rules for entering the date or time given in this section.

2

The MS-DOS File System

This chapter describes some facts about MS-DOS files, including:

- Types of files you can use with MS-DOS
- Conventions for naming files
- Rules for using wild-card characters
- How to create and run batch files
- The function of the AUTOEXEC.BAT file
- How to use peripheral devices (device files) for input/output

MS-DOS stores and processes data as files. A file is made up of one or more related characters; the capacity of a file is limited only by the data storage capacity of a diskette or fixed (hard) disk volume. MS-DOS uses files as locations for writing or reading data. A file's location can be a disk, or a peripheral device, such as the keyboard, screen, or printer. The data in most files has been input from the keyboard.

2.1 Types of Files

MS-DOS recognizes these types of files:

- System files
- Command files (utility programs)
- Data/text files
- Batch files
- Source files

System files are predefined programs that:

- Run the computer and its devices
- Control other programs loaded into memory from disk, such as word processing programs
- Process data entered from the keyboard

A **command file** is a predefined program on disk. You run the program by typing the filename, followed by optional parameters. The terms "external command" and "utility program" both refer to command files. Utilities or external commands perform basic file and diskette management tasks (such as formatting and copying diskettes) that are necessary for every application. Each utility program is stored on disk with a filename extension of .COM or .EXE.

A **data/text file** is a collection of alphanumeric characters that can be in document form, such as a memo. It can also be a list of data, such as a mail list, payroll statement, or the output of an application program, like VBASICA.

A **batch file** is a sequence of MS-DOS commands. When you load a batch file, it performs a specific task or tasks (see Chapter 2.5).

A **source file** contains English-like program language statements that you create. Your computer uses a translation program called a compiler to put each source statement line into a format that the machine can process.

2.2 Filename Conventions

The filename conventions described in this section are rules for specifying the name of a file, the filename extension, and the drive name. If you follow clear and simple file-naming conventions, you will be able to easily identify the contents of your files.

2.2.1 Filenames

Names for MS-DOS files can be one to eight alphanumeric characters. The characters you can use are:

- The letters A through Z (either upper- or lowercase)
- The numbers 0 through 9
- The special characters ! @ # $ % & ( ) - ' ^ { } ~ `

  Be careful when using some of these symbols in filenames, because you can reassign certain symbols to be switch characters. For more information on switch characters, see SWITCHAR in Chapter 6.3.1.

If you use any other characters, MS-DOS displays:

```
Invalid filename
```

Do not give any of your files the device names listed in Table 2-2.

2.2.2 Filename Extensions

You can add an optional filename extension to a filename. The extension is one to three alphanumeric characters preceded by a period. The valid characters for extensions are the same as for filenames.

You can use extensions to specify types of files. Table 2-1 lists some conventional MS-DOS file extensions and their meanings. Extensions other than those given in Table 2-1 or defined by your application program can be assigned at your discretion. For example, you might use your initials as the extension for the files you create, or the extension .TXT for text files or .MEM for memos.

If you specify more than three characters for the extension, MS-DOS accepts only the first three characters. If a data/text file has a filename extension, you must include the extension with the filename when you make a request for the file. To run a command or batch file, you omit the extension (.COM, .EXE, or .BAT) when you type the filename.

Table 2-1: MS-DOS File Extensions

FILE EXTENSION	MS-DOS INTERPRETATION
.ASM	8086 assembler language source code
.BAK	Backup file created by application
.BAS	BASIC source code (VBASICA)
.BAT	Batch command file
.COB	COBOL source code
.COM	Executable command file
.CRF	Cross-reference file
.DAT	Data file
.EXE	Relocatable executable file
.FOR	FORTRAN source code
.INT	Intermediate compiled code
.LIB	Library file
.LST	Listing of compilation or assembly
.MAP	Memory map from linker program
.OBJ	Relocatable object code module
.OVR	Overlay module
.PAS	Pascal source file
.PLM	PL/M source code
.PRN	Listing of compilation or assembly
.REF	Cross-reference listing
.$$$	Temporary system-generated file

2.2.3 Drive Names

The drive name is the letter name (A–O) of the disk drive where the file is currently located. Place a colon between the drive name and the filename itself. For example, to indicate that the file MAIL.LST is on drive B, type:

```
b:mail.lst
```

If you enter a filename without a drive name, MS-DOS searches for the file on the default drive.

To fully identify a file for MS-DOS, you might need to include the name of the subdirectory that contains the file. See Chapter 3 for information on using directory names.

The format for specifying a complete filename is:

```
drivename:filename.ext
```

The following description summarizes the characteristics of each part of the filename:

drivename:

- Is one letter (A–O)
- Is followed by a colon
- Does not need to be specified when the file is on the default drive

filename

- Contains up to 8 characters
- Does not use the symbols " / [ ] + = , ; : | < > \ (sp)

.ext

- Is preceded by a period (.)
- Contains up to 3 characters
- Does not use " / [ ] + = , ; : | < > \ (sp)
- Is optional except for some of the extensions in Table 2-1

2.3 Filename Wild-Card Characters

The ? and * wild-card characters replace existing characters in a filename. You can use wild-card characters to match a specified portion of a filename with existing filenames.

Wild-card characters can be used with these commands:

COPY	ERASE
DEL	PRINT
DIR	REN

The ? wild-card character used in a filename with an MS-DOS command matches any single existing character (or none) in that position.

For example, suppose your disk directory contains several similar filenames, such as:

CHAPT1
CHAPT2
CHAPT3

If you enter this command:

dir chapt?(cr)

MS-DOS displays only the filenames that match the first five characters of the filename in the command. The sixth character can be any character, as indicated by the ?.

The * wild-card character replaces any existing character in that position and all subsequent characters. A matching filename can be 2 to 8 characters long, and the other characters in the filename can be any character.

For example, if you have the sample directory shown above and you enter the following command, MS-DOS displays all the filenames that start with CH:

```
dir ch*(cr)
```

MS-DOS also displays any other filenames in your directory that begin with CH, such as CHARLES.MEM, CHART.DAT, or CHECKS.

You can also use the * to display all the files ending with a particular extension. For example, if you enter the following command:

```
dir *.bat(cr)
```

MS-DOS lists the names of all the batch files (indicated by .BAT) in the current directory.

You can specify *.* in a command to indicate all the filenames in a directory. For example, using *.* with the DEL command deletes all the files from the directory. **Note:** When you specify DEL *.*, MS-DOS asks if you are sure before it deletes the files. Make sure that you have backed up any files you need.

You can use ? and * together in a command. For example, if your directory contains:

83JOBS.PUB
82JOBS.PUB
83JOBS.ACT

You can enter this command:

```
dir 8?jobs.p*(cr)
```

MS-DOS displays these filenames:

```
83JOBS.PUB
82JOBS.PUB
```


83JOBS.ACT is not displayed, because .A does not match .P.

2.4 Device Names

MS-DOS can send data to or receive data from a device. MS-DOS interprets device names as the names of peripheral devices. You can use these device names as parameters in MS-DOS commands to refer to devices as if they were files. A device name, however, never has a file extension or drive name because device names are not disk files, but represent input/output devices. "Logical device" is another term for a device name.

The device names listed in Table 2-2 include the logical device names for the standard MS-DOS device drivers (programs that control the peripheral devices connected to your computer). Device drivers are contained on the system diskette. At bootup, drivers are loaded by the ICONFIG.SYS/VCONFIG.SYS file, and the drivers' names (logical device names) are displayed on the screen.

Table 2-2: MS-DOS Device Names

NAME	DEVICE TYPE
AUX COM1	Names for the device driver for serial port 1.
COM2	The name for the device driver for serial port 2.
CON	Keyboard input to the CPU, and CPU output to the screen.
LPT1 LST PRN	Names for the device driver for parallel port 1.
LPT2	The name for the device driver for parallel port 2 (reserved, but not in standard hardware).
LPT3	The name for the device driver for parallel port 3 (reserved, but not in standard hardware).
NUL	A nonexistent device for a command that requires a filename. Use this device when you do not want to create a file. You can also use it to get rid of unwanted console output.

For example, you can copy data from the CON device file (the keyboard) to a file on disk called PHN.LST:

```
copy con phn.lst(cr)
```

The next lines you type after the COPY CON command are temporarily stored in memory until you type CTRL-Z (ALT-Z for V mode) and press the Return key. Then the text you have entered is stored in the file PHN.LST.

You can also redirect output to a device by using the > symbol (described in Chapter 5). For example, this command sends the directory to the printer (PRN):

```
dir > prn(cr)
```

2.5 Batch Files

A batch file is a sequence of MS-DOS commands that perform a specific task. You can create batch files with the line editor EDLIN, with the COPY command (described in Chapter 6), or with any word processing program. You must give batch files the extension .BAT. To run the sequence of commands stored in a batch file, type the name of the file without the extension.

Using batch files, you can execute several MS-DOS commands by entering a single batch command. Because batch files can perform many different functions, writing batch files is similar to programming, and the files themselves are like short programs "written in MS-DOS." This chapter gives the basic information you need to start using and writing batch files; you will find examples of batch file usage throughout this manual.

The following example is a batch file named NEWDISK.BAT that contains a sequence of commands to display the directory of a formatted disk and check the diskette for inconsistencies. It also displays remarks (lines beginning with REM) that describe the batch file.

```
REM This is a file to check formatted disks
REM It is named NEWDISK.BAT
PAUSE Insert disk in drive B
DIR B:
CHKDSK B:
```

Once the NEWDISK.BAT file is saved on disk, you can run the two commands (DIR B: and CHKDSK B:) by typing the batch file name NEWDISK. PAUSE and REM are commands used only for batch files. These commands and the other commands used for batch files (FOR, IF, GOTO, ECHO, and SHIFT) are described in Chapter 6.4.

2.5.1 Batch File Conventions

Follow these rules when you use batch files:

1. To run a batch file, specify the batch file name (without the .BAT extension). If the batch file contains replaceable parameters, follow the filename with the values you want to substitute for the parameters (see Chapter 2.5.2).
2. If you press CTRL-C (ALT-C in V mode) while a batch file is being processed, MS-DOS displays:

```
Terminate batch job (Y/N)?_
```

   Type Y if you want MS-DOS to ignore the rest of the commands in the batch file and return you to the system prompt. Type N to stop the current command being processed; MS-DOS then processes the next command in the batch file.
3. You can specify the name of another batch file as a command in the batch file. MS-DOS invokes that batch file from the one being processed.

2.5.2 Replaceable Parameters

Replaceable parameters are "dummy values" in a batch file. They are numbers preceded by a percent sign (such as %1 and %2). When you run a batch file, you include the value(s) you want to use in place of the replaceable parameters in the file.

You can specify ten replaceable parameters in a batch file (%0 to %9). %0 is always replaced by the drive name (if one is designated) and the name of the batch file. %1 is replaced by the first value following the filename in the command you enter to run the batch file. %2 is replaced by the second value, and so on. You can specify more than ten replaceable parameters by using the SHIFT command.

For example, you can use the COPY CON command to create a phone list file:

```
copy con phn.lst(cr)
Mary Smith (415) 123-4567(cr)
Engineering (408) 111-1111(cr)
George Jones (916) 555-5555(cr)
Marketing (408) 777-7000(cr)
^Z(cr)
```


Then you can create a batch file named LOOKING that uses the FIND command to look through PHN.LST for a dummy value (the replaceable parameter %1):

```
copy con looking.bat(cr)
find "%1" phn.lst(cr)
^Z(cr)
```

When you load the batch file, include the value or name that you want to substitute for the replaceable parameter in the batch file. For example, if you specify:

```
looking 408(cr)
```

MS-DOS replaces the parameter %1 with 408. MS-DOS then processes the LOOKING batch file using this value and displays the command from the file:

```
A>find "408" phn.lst
```

The FIND command then displays all the lines from the PHN.LST file that contain 408:

```
---------- phn.lst
Engineering (408) 111-1111
Marketing (408) 777-7000
```


2.5.3 The AUTOEXEC.BAT File

AUTOEXEC.BAT is the name of a batch file that MS-DOS always looks for and processes immediately after it loads into memory. If MS-DOS does not find an AUTOEXEC.BAT file, MS-DOS prompts you for the date and time and then displays its command prompt. When there is an AUTOEXEC.BAT file, MS-DOS displays the date and time prompts only if the DATE and TIME commands (described in Chapter 6.4) are included in the AUTOEXEC.BAT file.

Create an AUTOEXEC.BAT file if you want a command or sequence of commands to operate each time MS-DOS loads. For example, suppose you are in V mode and your principal application is VBASICA. You can use COPY CON to create an AUTOEXEC.BAT file that takes you straight into VBASICA when you boot up:

```
copy con autoexec.bat(cr)
basica(cr)
^Z(cr)
```

The command "BASICA" tells MS-DOS to load the VBASICA language into memory. If this AUTOEXEC.BAT file and the command file are on your system diskette, then each time MS-DOS loads into memory it displays the first command from the file:

```
A>basica
```


VBASICA loads into memory and displays its command prompt (Ok). You can now use VBASICA. To return to MS-DOS, use the VBASICA command "SYSTEM" to exit to the operating system.

If you create an AUTOEXEC.BAT file and want to process it immediately, enter this command:

autoexec(cr)

or reload MS-DOS either by pressing the reset button or by pressing the CTRL, ALT, and DEL keys at the same time.

3

The MS-DOS Subdirectory System

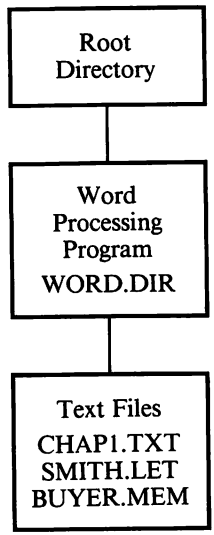
MS-DOS allows you to create and use a hierarchical subdirectory system. This chapter describes the subdirectory system and how to work with it. The first topic is a general introduction, followed by the rules for using subdirectory names to locate files. Then this chapter describes the MS-DOS commands used with subdirectories:

- DIR displays the current directory.
- MKDIR creates a subdirectory.
- CHDIR changes from one directory to another.
- RMDIR deletes an empty subdirectory.

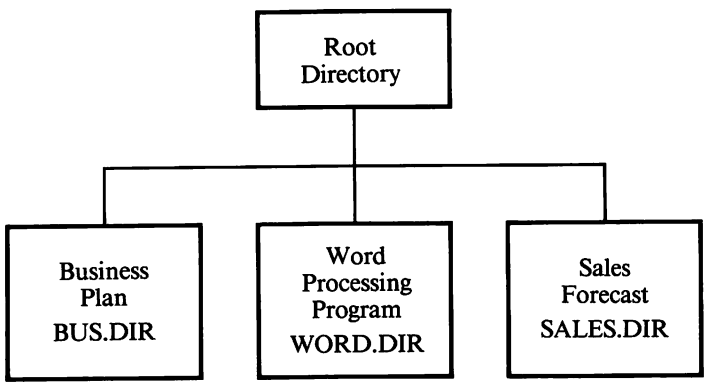
A **directory** is a list of the files on a diskette, including the filenames, the file size, and the date and time each file was created. MS-DOS enters the filename and date and time in the directory each time you create or update a file.

MS-DOS has a main directory to which you can add subdirectories. Subdirectories are hierarchical, like a family tree or an organization chart. The main or "root" directory is created by the FORMAT program during a disk format. The root directory is the start of your subdirectory system. The root directory contains files and can also contain the names of subdirectories. Each subdirectory in turn can contain files and other subdirectories.

For example, you can create a subdirectory called WORD.DIR and place your word processing application program in that subdirectory. Then you can store all the text files you create in the word processing subdirectory, WORD.DIR, as shown in the following diagram.



Similarly, you can create a different subdirectory for each application program, and group your files in the appropriate subdirectory:



When you save a file, MS-DOS puts the filename in the directory you are currently using or in another directory that you specify. A file in one directory can have the same name as a file in another directory because MS-DOS keeps each directory separate.

You can search for a file in one subdirectory rather than searching through many kinds of files in one large main directory. Chapter 3.1 describes how to specify the subdirectory name (or path) for a file.

3.1 Using Directory Names (Paths) to Locate Files

When you use hierarchical directories, you must tell MS-DOS where to find your files in the directory structure. In addition to the filename, you give the name of the subdirectory where the file is located.

Giving the subdirectory name means specifying a **path** for MS-DOS to follow through the directory hierarchy. A path or subdirectory name:

- Is a sequence of directory names, each separated from the previous one by the path-separator character, usually a backslash (\). (The path-separator character becomes a slash (/) if you change the switch character to any character other than a slash. For more information, see the SWITCHAR command in Chapter 6.3.1.)
- Leads MS-DOS through the directory hierarchy. Each \ leads to a lower subdirectory level.
- Can end with either a filename or a directory name.
- Can go through only existing subdirectories.
- Must not exceed 63 characters.

The format for a path to a file or subdirectory is:

[drivename:][[\]directoryname[\directoryname...]\][filename.ext]

drivename
is the drive containing the file or directory you want. You can omit the drive name if the file or directory is on the default drive.

\
The initial backslash always indicates the root directory. If a path begins with a backslash, MS-DOS starts searching for the file or directory at the root directory. Otherwise, MS-DOS searches downward from the current directory.

directoryname
is the name of a subdirectory (created with the MKDIR command, described in Chapter 3.3). If the first subdirectory name is not preceded by a backslash, it is directly below the current directory. Each additional \directoryname indicates another subdirectory level below the root directory or below the current directory.

filename.ext
is the full filename and extension of the file you want. The filename is always last in the path. If the file is in the current directory, you do not need to specify a path with the filename.

MS-DOS recognizes three abbreviated pathnames:

.
is shorthand for the current directory. You can use the . notation with the DIR command to display the current subdirectory, although it is not necessary.

..
is shorthand for the current directory's parent directory. MS-DOS uses the . and .. directories to create the hierarchical tree directory structure. You can use the .. notation with DIR, CHDIR, and DEL.

\
is shorthand for the root directory. You can use the \ notation with the CHDIR command.

3.1.1 Examples of Paths

The following examples demonstrate different kinds of pathnames.

```
\WORD.DIR
\MGR1.NAM
```

These are subdirectory names. The initial \ indicates the root directory. Both WORD.DIR and MGR1.NAM are first-level subdirectories (directly below the root directory).

```
\WORD.DIR\CHAP1.TXT
\MGR1.NAM\JOBS\CLERK
```

These are full pathnames. In the first example, the file or subdirectory CHAP1.TXT is in the first-level subdirectory WORD.DIR. In the second example, the file or subdirectory CLERK is in the subdirectory JOBS under the first-level subdirectory MGR1.NAM.

```
WORD.DIR\DAVID
JOBS\BENEFITS
```

These are relative pathnames that refer to the current directory level. In the first example, DAVID is a file or directory in the subdirectory WORD.DIR below the current directory. In the second example, the file or directory BENEFITS is in the JOBS subdirectory, which is listed in the current directory. If the current directory is the root directory, these pathnames are the same as specifying \WORD.DIR\DAVID and \JOBS\BENEFITS.

3.1.2 Working with Files in Subdirectories

You can use paths or subdirectory names with any of the MS-DOS commands that operate on filenames. This section gives several examples of using pathnames in MS-DOS commands.

You locate a file by specifying a path for MS-DOS to follow through the directory hierarchy. The filename is always the last name in the path. When the file is in the current directory, you can omit the path and give just the filename. To get to a file in the previous directory, you can use .. as the path.

Displaying Files Onscreen

You can display the contents of text files on your screen using the TYPE command. For example, to display the CHAP1.TXT file in the WORD.DIR subdirectory on the default drive, enter:

```
type \word.dir\chap1.txt(cr)
```

In the preceding example, the first \ tells MS-DOS to start the search for the file at the root directory. MS-DOS finds the subdirectory WORD.DIR listed in the root directory. In the WORD.DIR subdirectory, MS-DOS finds the CHAP1.TXT file and displays the file on the screen.

If WORD.DIR is the current directory, you can specify the filename without a path, like this:

```
type chap1.txt(cr)
```

If you include the drive name in a command, the drive name must precede the path. For example, this command displays CHAP1.TXT from the WORD.DIR subdirectory on drive B:

```
type b:\word.dir\chap1.txt(cr)
```

Copying Files

You use the internal command COPY to copy files. To copy a file from one directory to another, you must specify the path to the file to be copied and the path to the directory where the file is to be copied. For example, to copy the CHAP1.TXT file from the WORD.DIR subdirectory on drive A to the root directory on drive B, and to rename the file INTRO, enter:

```
copy a:\word.dir\chap1.txt b:\intro(cr)
```

Deleting Files

To delete files from a directory, you use the internal command DEL (Delete). If the file you want to delete is in the current directory, you do not need to specify a path to the file. For example, if you are using the WORD.DIR directory, you can delete a file from that directory with the command:

```
del chap1.txt(cr)
```

If you are using any other directory, you must include the path to the file you want to delete from WORD.DIR. For example, this command deletes CHAP1.TXT from the WORD.DIR subdirectory on drive B:

```
del b:\word.dir\chap1.txt(cr)
```

You can use the .. notation with DEL to refer to the previous directory. For example, this command deletes MYFILE from the parent directory of the current directory:

```
del ..\myfile(cr)
```

3.1.3 Paths and External Commands

When you enter a command at the system prompt, the PATH command searches your directories for the program file. PATH searches only for .EXE, .COM, and .BAT files. Text files and other types of files cannot be accessed using the PATH command.

You do not need to move to the subdirectory where the program files are contained. Instead, a path has been provided on your system diskette to the subdirectories where the correct program files are located. See Chapter 1.1.1 for an explanation of how the PlusPC MS-DOS system diskette stores program files.

Whenever you enter an external command, MS-DOS searches the current directory and then the subdirectory that was specified with the PATH command. If you specify PATH while in one mode, you still need to specify another PATH when you enter the other mode. If you reboot to the same mode, or switch to the other mode, you will have to reset PATH (see the PATH command in Chapter 6.4).

3.2 Listing a Directory

The DIR command, described in Chapter 6.4, lists the contents of a directory. You can list a directory onscreen or send it to a device or file. To list a subdirectory with DIR, you can use the path notation shown in Chapter 3.1.1. For example, you can display the current directory on the default drive:

```
dir(cr)
```

To display the parent directory (one level above the current directory), you can use the shorthand notation:

```
dir ..(cr)
```

If you enter the DIR command from the WORD.DIR subdirectory, MS-DOS displays that subdirectory. To display the WORD.DIR subdirectory on the default drive (regardless of the directory you are in), enter:

dir \word.dir(cr)

The display for the WORD.DIR subdirectory might look like this:

```
Volume in drive A has no ID
Diskette is V format
Directory of A:\word.dir

.            <DIR>          5-05-85  10:09a
..           <DIR>          5-05-85   2:15p
DAVID.L      <DIR>          5-20-85   3:40p
CHAP1.TXT              3345  6-15-85  11:13a
       4 File(s)   370688 bytes free
```


This DIR display gives the following information:

- No Volume ID was assigned to the diskette during formatting or with the VOL command.
- The diskette is V (VICTOR) format.
- This directory is WORD.DIR, a first-level subdirectory.
- The single period (.) indicates the current directory. The . is displayed only for subdirectories, not for the root directory. MS-DOS automatically creates the . entry when you create a directory.
- Two periods (..) indicates the parent directory (the subdirectory level above this one).
- The WORD.DIR subdirectory has another subdirectory (DAVID.L) below it.
- WORD.DIR contains the file CHAP1.TXT, which is 3345 bytes, and was created or last modified at 11:13 am on June 15, 1985.

3.3 Making a Subdirectory

You can create a subdirectory by entering a MKDIR (Make Directory) command from the keyboard or from a batch file. See the descriptions of the MKDIR and FOR commands in Chapter 6.4. You can shorten MKDIR to MD. In the MKDIR command, give the name of the directory you want to create. MS-DOS creates a subdirectory as a special kind of file with a < DIR > attribute.

The conventions for naming a directory are the same as for naming a file. You can use 1 to 8 alphanumeric characters, and you can include an optional filename extension of 1 to 3 alphanumeric characters preceded by a period.

For example, suppose you want to create a subdirectory named MGR1.NAM under the root directory on the diskette in the default drive. If you are in the root directory, you can enter:

```
mkdir mgr1.nam(cr)
```

If you are in another subdirectory, you can enter:

```
md \mgr1.nam(cr)
```

You can verify that MS-DOS created the subdirectory by entering the DIR command.

Once a directory exists, you can add other subdirectories to it. For example, you can create a subdirectory called JOBS beneath MGR1.NAM. Enter:

```
md \mgr1.nam\jobs(cr)
```

You can also create a subdirectory by first moving into the parent directory, and then making the subdirectory. You move from one directory to another with the CHDIR command, described in Chapter 3.4. For example, enter these commands:

```
chdir mgr1.nam(cr)
mkdir jobs(cr)
dir(cr)
```

MS-DOS moves to the MGR1.NAM subdirectory, creates the JOBS directory, and displays the current directory, MGR1.NAM:

```
Volume in drive A has no ID
Diskette is V format
Directory of A:\mgrl.nam

.            <DIR>      6-30-85      9:00a
..           <DIR>      8-09-85     10:09a
JOBS         <DIR>      8-09-85     10:34a
        3 File(s)   370688 bytes free
```

If you create and save a file now, MS-DOS places it in the MGR1.NAM subdirectory. For example, if you enter this COPY CON command, MS-DOS creates and saves a file named TITLES in the MGR1.NAM subdirectory:

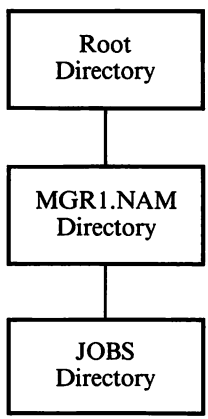
```
copy con titles(cr)
Publication Manager(cr)
Managing Editor(cr)
Copy Editor(cr)
^Z(cr)
```

Now the name of the TITLES file, its size, and the date and time it was created are displayed in the MGR1.NAM directory:

```
Volume in drive A has no ID
Diskette is V format
Directory of A:\mgr1.nam

.             <DIR>        6-30-85     9:00a
..            <DIR>        8-09-85    10:09a
JOBS          <DIR>        8-09-85    10:34a
TITLES                51   8-09-85    11:20a
        4 File(s)   370637 bytes free
```


The full path to JOBS is \MGR1.NAM\JOBS, and the path to the TITLES file is \MGR1.NAM\TITLES. Each \ indicates a level below the root directory. MGR1.NAM is the first subdirectory (indicated in the directory by the .), and JOBS is the subdirectory beneath it:

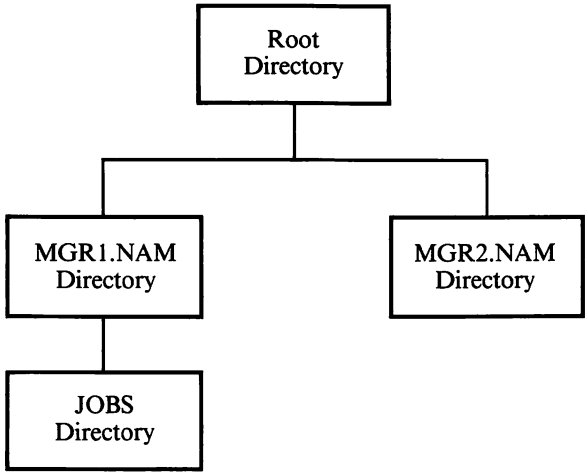


To add another subdirectory, such as MGR2.NAM, at the same level as MGR1.NAM, you must tell MS-DOS to start at the root directory:

mkdir \mgr2.nam(cr)

The first \ in a MKDIR command indicates the root directory to MS-DOS. You can omit the initial \ if you are at the root directory.

MS-DOS creates MGR2.NAM and puts the new directory name in the root directory:



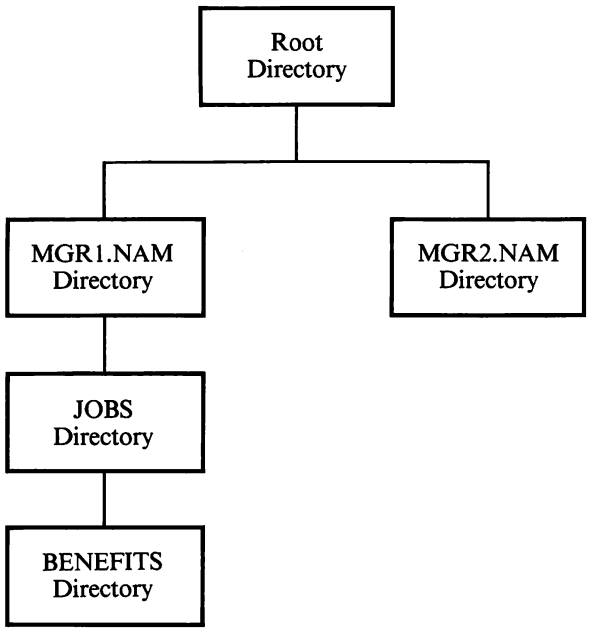
You can add a third directory level beneath JOBS, such as a subdirectory named BENEFITS. If you are currently at the JOBS subdirectory, you can create BENEFITS by entering:

```
mkdir benefits(cr)
```

If you are not at JOBS, enter:

```
mkdir \mgr1.nam\jobs\benefits(cr)
```

Now you have the following subdirectory system:



Because each subdirectory name is listed in the preceding directory, you cannot create a new subdirectory unless the parent directory exists. If you load MS-DOS and try to create several new subdirectories in the same command, such as:

```
mkdir mgr2.nam\mgr3.nam\mgr4.nam(cr)
```

MS-DOS responds with:

```
Unable to create directory
A>_
```

Likewise, MS-DOS does not allow you to give a file the same name as its subdirectory. This is because subdirectories are listed as files in a directory, and duplicate filenames are not allowed.

3.4 Moving into a Directory

Using a hierarchical file structure gives you two means of locating files. You can enter a command in any directory and locate the file by using a pathname, or you can move to the directory in which the file is located and enter the command there.

You move into a directory to:

- Display the directory
- Create files or subdirectories
- Delete files
- Use files or programs in the directory
- Remove subdirectories

The CHDIR (Change Directory) command tells MS-DOS to move from one directory to another. You can shorten CHDIR to CD.

If you enter the CHDIR command without any parameters, MS-DOS displays the name of the current directory. Entering CHDIR with a subdirectory name moves you to that directory if it is listed in the current directory. If you enter CHDIR and a subdirectory name that is not listed in the current directory, MS-DOS displays:

```
Invalid directory
```

Using the examples in Chapter 3.3, if you are in the root directory, you can move into the next subdirectory (MGR1.NAM) by entering one of the following commands:

```
chdir mgr1.nam(cr)
cd \mgr1.nam(cr)
```

The first \ in a CHDIR command indicates the root directory. If a directory name follows the backslash, it names the first subdirectory below the root directory. Each additional \ in the command is another subdirectory level in the path to the specified directory. The command:


```
chdir \mgr1.nam\jobs\benefits(cr)
```

places you in the BENEFITS subdirectory. To move up to the MGR1.NAM subdirectory, enter:

```
chdir \mgr1.nam(cr)
```

To get to JOBS from MGR1.NAM (now the current directory), enter:

```
chdir jobs(cr)
```

If you are not at MGR1.NAM, you must indicate the path to JOBS:

```
chdir \mgr1.nam\jobs(cr)
```

You can move to the level above the current directory level by specifying .. with CHDIR:

```
chdir ..(cr)
```

This command moves you to the directory above the one you are now using. For example, if you are currently at JOBS, MS-DOS moves up to MGR1.NAM.

Entering CHDIR followed by a \ always returns you to the root directory. You should enter this command when you finish working in a subdirectory; if you do not, you might save files in the wrong subdirectory. CHDIR \ ensures that MS-DOS is at the root directory.

3.5 Removing a Subdirectory

The RMDIR (Remove Directory) command removes an empty subdirectory from the hierarchy. You can abbreviate RMDIR to RD.

Before you can remove a subdirectory, you must delete all filenames (with DEL) and any lower subdirectories (with RMDIR). An empty directory lists only the . and .. notations, which cannot be deleted because they are part of the MS-DOS hierarchy system.

You can remove only one subdirectory at a time. For example, if you want to remove the BENEFITS subdirectory, first delete any files in BENEFITS. If you are in the BENEFITS subdirectory, use CHDIR to move to another directory. To remove the BENEFITS subdirectory if you are in the JOBS subdirectory, enter:


```
rmdir benefits(cr)
```

You can also specify a full path from the root directory. For example, this command removes the BENEFITS subdirectory from the JOBS subdirectory if BENEFITS is already empty:

```
rmdir \mgr1.nam\jobs\benefits(cr)
```

If you are currently using the MGR1.NAM directory, you can specify the path to the empty BENEFITS subdirectory and remove it:

```
rmdir jobs\benefits(cr)
```

If the directory you name does not exist, or if you try to remove the current directory or one that is not empty, MS-DOS displays:

```
Invalid path, not directory,
or directory not empty
```

4

Using the Line Editor (EDLIN)

EDLIN is a line editor program used mainly to create and edit source files (such as MS-DOS batch files). You can also use EDLIN to create or edit text. To input text such as memos or documents, however, it is simpler and more efficient to use a word processing program.

You can use two types of commands for editing with EDLIN:

- File commands, which create or edit files
- Line commands, which edit characters within a single line of a file

The file commands are unique to EDLIN. The line commands are the MS-DOS command-line editing functions (described in Chapter 5); these differ depending on the mode you are using.

To use EDLIN, load the program as shown in Chapter 4.1. Chapter 4.1.1 shows you how to create a batch file; this batch file is used in examples throughout the chapter. Chapter 4.1.2 shows you how to edit an existing file. Chapter 4.2 discusses how to use the EDLIN file commands and introduces the format of the commands. Chapter 4.2.3 summarizes the EDLIN file commands.

4.1 Invoking EDLIN

The EDLIN.COM file must be available on disk in order to load the program. If you are using subdirectories, MS-DOS must be able to find EDLIN.COM in the directory system. If it is not in the current subdirectory, give its subdirectory location in the EDLIN command, or enter a PATH command to indicate its location.

You load EDLIN and the file you want to edit at the same time:

edlin filename(cr)

The filename can be either the name of an existing file or the name of the file you are creating. Include the drive name if the file is not on the default drive. To edit a data/text file in a subdirectory, either change to that subdirectory before loading EDLIN or give the subdirectory name with the filename when you invoke EDLIN. Chapter 3 describes how to use the MS-DOS subdirectory system.

4.1.1 Creating a File

To create a new file, enter EDLIN followed by the name of a file that does not exist. Include the drive on which the file is to be saved unless you want to save it on the default drive.

For example, to create the file NEWDISK.BAT in subdirectory MGR1.NAM on the diskette in drive B, enter:

edlin b:\mgr1.nam\newdisk.bat(cr)

EDLIN creates a new, empty file and displays the message "New file" and the command prompt (*):

```
New file
*_
```

To enter text into the file, type the Insert command (I) at the * prompt and press Return. EDLIN then numbers and displays the first line for your entry:

```
New file
*I(cr)
        1:*_
```

Now you can type in the text of the file. End each line with a Return. EDLIN then displays the next line number.

Here is an example of an MS-DOS batch file created with EDLIN. This file displays the directory of a formatted disk and checks the disk directories for consistency.


```
A>edlin b:\mgrl.nam\newdisk.bat(cr)
New file
*I(cr)
    1:REM This is a file to check formatted
      disks(cr)
    2:REM It is named NEWDISK.BAT(cr)
    3:PAUSE Insert formatted disk in drive B(cr)
    4:DIR B:(cr)
    5:CHKDSK B:(cr)
```

You can stop text entry without saving any of the file by entering the Quit (Q) command. When you finish entering data, type a CTRL-C (Interrupt) or CTRL-Z (Terminate Text Entry command) on the next line and press Return. Either CTRL-C or CTRL-Z stops Insert mode and returns you to the * prompt. (If you are using EDLIN in V mode, substitute ALT-C for CTRL-C, and ALT-Z for CTRL-Z.)

To display the file you created, enter the List command (L). EDLIN displays the file and numbers the lines. (Line numbers are not saved in the file on disk.) When you insert new lines or delete existing lines, EDLIN adjusts the line numbers accordingly. The line numbers always run from 1 through the number of the last line.

To save the file on disk, type the End command (E). You are returned to the MS-DOS command prompt.

4.1.2 Editing an Existing File

To edit an existing file, specify EDLIN, the drive and subdirectory containing the file, and the filename with its extension. For example, if you want to make changes to the file NOTICE.MEM in the WORD.DIR subdirectory on the default drive, enter:

```
edlin \word.dir\notice.mem(cr)
```

EDLIN loads the file into memory. If EDLIN can load the entire file, it displays the following message and the asterisk command prompt (*):

```
End of input file
*_
```

If EDLIN cannot load the entire file, it fills up 3/4 of available memory with the first part of the file and does not display the "End of input file" message.

To edit a file larger than memory:

1. Display the portion of the file in memory with the List Line command (L, described in Chapter 4.2.3).
2. Edit the portion of the file in memory.
3. Save the edited portion of the file on disk with the Write command (W, described in Chapter 4.2.3). Add the rest of the lines from the file by using the Append command (A, described in Chapter 4.2.3).

4.2 Using File Commands

Use the Insert command (I) to create an EDLIN file, as described in Chapter 4.1.1. After the file is created, you can use the EDLIN file commands to:

- Add lines from disk to memory if the file is too large for memory (Append).
- Delete one or more lines from the file (Delete).
- Modify existing lines (Edit).
- Add one or more lines of text to a file (Insert).
- Display the series of lines that you want to change (List).
- Replace previous text with new text within a certain number of lines in the file (Replace).
- Search through a specified number of lines in a file for a string of characters that you want to change (Search).
- Write edited portions of a large file from memory to disk (Write).

File commands are single letters (such as A for Append) with optional parameters. The parameters select, display, or change one or more lines in a file. You type the letter of the command and the parameters you want from the keyboard. The parameters and the commands are summarized in Table 4-1 and in Chapter 4.2.3.

4.2.1 File Command Parameters

The file command parameters are shown in Table 4-1.

Table 4-1: EDLIN File Commands and Parameters

COMMAND PARAMETER	PARAMETER DESCRIPTION	COMMANDS THAT USE PARAMETER
n	Number of lines.	
line	Represents the line number you select. For the line number, you can specify a decimal integer from 1–65534. If the number is larger than the lines in memory, the number indicates the line after the last existing line. Line numbers in a command must be separated from each other by a comma or a space. Lines can also be specified by:	A—Append lines W—Write lines
	. (a period) represents the current line (marked on the screen by an * between the line number and the first character of the line).	D—Delete lines N—Edit lines I—Insert lines L—List text R—Replace a string of characters
	# (the pound sign) specifies the line after the last line (same as specifying the number larger than the last line number in the file).	
	(cr) (Return) directs EDLIN to use a default line number appropriate to the command.	
?	Directs EDLIN to query for a yes or no response to an Ok? prompt.	R—Replace a string of characters S—Search for a string
string	Represents one or more characters to be found or replaced. Each string must end with a CTRL-Z or a Return. There should be no spaces between the string parameters unless the space is part of the string. Do not enter spaces between string parameters and the command.	R—Replace a string of characters S—Search for a string in a line

4.2.2 File Command Conventions

Some conventions are common to all the file commands. With any file command you can:

- Type commands with or without a space between the line number and the command. For example:

  5I

  can also be typed:

  5 I

- Reference lines relative to the current line.

  On the screen, the current line has an * between the line number and the first character of the line. You include a plus sign with a line number in a command to indicate the number of lines following the current line. A minus sign with a line number in a command indicates the number of lines before the current line.

  For example, this command directs EDLIN to begin inserting text one line before the current line on the screen:

  −1I

- Issue multiple commands on one command line. Separate multiple commands in a line with a semicolon. For example, if you want to edit a line and then display other lines, enter:

  10; − 2, + 1L(cr)

  This command lets you edit line 10, then display (List) the two lines before line 10 and the line following line 10. The comma after − 2 is a syntax element for "line,line"—see the List command in Chapter 4.2.3.

  If you are issuing a multiple command with a string parameter, as in a Search or Replace command, separate the commands with a CTRL-Z.

For example, this command:

2,10 SThis string^Z – 2; + 2L(cr)

searches for the first occurrence of the characters "This string" in lines 2 through 12. It then displays the 2 lines before the line that contains "This string" and the 2 lines after it.

If "This string" cannot be found, the 2 lines before the current line (noted on the screen with the *) and the 2 lines after the current line are displayed.

4.2.3 Descriptions of File Commands

4

This section presents detailed descriptions of each EDLIN command for editing files. The descriptions are given alphabetically.

Append Lines (A)

[n]A(cr)

A adds the number of lines, specified by n, from disk to the end of the lines currently in memory.

n represents the number of lines.

Use this command for large files that cannot fit into memory at one time. Edit the lines in memory, write them to disk with the Write command (W), then use the Append command (A) to put the rest of the file in memory.

If you enter a number with the A command, that number of lines is appended to the part of the file currently in memory. If no number is entered, the lines are read into memory until memory is 3/4 full. If memory is already full, no more lines can be appended.

When the last line of the file is in memory, EDLIN displays:

```
End of Input File
*_
```

Delete Lines (D)

[line][,line]D(cr)

D deletes the lines specified. "Line" can be any of the parameters for a line described in Table 4-1. Without parameters, D deletes the current line.

line
deletes the specified line.

,line or ,#
deletes the current line (marked by the *), the specified line, and all lines between the current line and the specified line.

line, or line,line or line,#
deletes the specified lines and all lines between them.

The line immediately after the deleted lines becomes the current line (marked by the *).

For example, to delete lines 2 and 3 in the NEWDISK.BAT file:

```
1:REM This is a file to check formatted disks
2:REM This is a batch file created by EDLIN
3:REM This file is named NEWDISK.BAT
4:PAUSE Insert formatted disk in drive B
5:DIR B:
6:CHKDSK B:
```

Enter:

2,3 D(cr)

To verify the result, specify the List command (L):

```
*L(cr)
1:REM This is a file to check formatted disks
2:*PAUSE Insert formatted disk in drive B
3:DIR B:
4:CHKDSK B:
```


After lines 2 and 3 are deleted, line 4 becomes line 2 and is marked by the asterisk as the current line.

To delete a single line, such as line 3, enter:

3D(cr)

To see the result, specify the List command (L):

```
*L(cr)
1:REM This is a file to check formatted disks
2:PAUSE Insert formatted disk in drive B
3:*CHKDSK B:
```

When line 3 is deleted, line 4 becomes line 3 and is marked as the current line.

To delete the current line, now marked as line 3, enter:

D(cr)

Specify the L command at the * prompt, and the result is:

```
1:REM This is a file to check formatted disks
2:PAUSE Insert formatted disk in drive B
```

Edit Line

[n](cr)
[.](cr)
[#](cr)

n represents the line number.

. indicates the current line.

# indicates the line after the last line in the file.

See Table 4-1 for an explanation of the line number parameters.

EDLIN displays the line number and the text on that line. On the next line, it repeats the line number and displays the asterisk prompt and the cursor. You can start editing the line at the cursor position. You can use any of the command-line editing keys described in Chapter 5 to edit the line. Note that the command-line editing keys are different in V mode and I mode.

If you do not want to change the current line (marked by the *), and the cursor is at the beginning of the line, press the Return key to accept the line as it is. If you press Return while the cursor is in the middle of the line, the rest of the line is deleted.

For example, you can select line 1 with the Edit Line command:

```
*1(cr)
```

EDLIN displays the specified line:

```
1:*REM This is a file to check formatted disks
1:*_
```


You can now type in a new line 1, or you can use the line commands to edit the current line (the template, described in Chapter 5).

End Edit (E)

E(cr)

E ends the editing session, saves the edited file on disk, changes the original file's extension to .BAK, and exits to MS-DOS. If the file was newly created during the editing session, no .BAK file is created.

Because the E command takes no parameters, you cannot select the drive on which to save the file. If you did not designate the drive when you invoked EDLIN, such as EDLIN B:NEWDISK, the file is saved on the default drive.

If the disk does not have enough free space for the entire file to be written, the write is aborted and the edited file is lost. Part of the file, however, may be written to the disk.

Insert Lines of Text (I)

[line]I(cr)

I inserts line(s) of text immediately before the specified line.

"Line" can be any of the parameters for a line described in Table 4-1.

You must enter the I command to create a new file. The first insert begins with line number 1. EDLIN numbers each new line sequentially each time you press Return. EDLIN remains in insert mode until you enter a CTRL-Z or a CTRL-C as your last line (ended with a carriage return).

If you insert lines between existing lines in a file, all line numbers following the inserted section are incremented by the number of lines inserted.

If you do not include a line number, the new lines are inserted immediately before the current line. If the specified line number is larger than the last line number or if # is specified as the line number, the inserted lines are appended to the end of the file. In this case, the last line inserted becomes the current line.

For example, to add lines before line 2 in the following file:

```
1:REM This is a file to check formatted disks
2:PAUSE Insert formatted disk in drive B
3:DIR B:
4:CHKDSK B:
```

Enter:

```
2 I(cr)
```

EDLIN displays:

```
2:*_
```

Enter:

```
2:REM This file is a batch file(cr)
3:REM This batch file is called NEWDISK(cr)
4:^Z(cr)
```


You can display the new file with the L command:

```
1:REM This is a file to check formatted disks
2:REM This file is a batch file
3:REM This batch file is called NEWDISK
4:*PAUSE Insert formatted disk in drive B
5:DIR B:
6:CHKDSK B:
```

To insert lines immediately before the current line (line 4 in the previous example), enter:

I(cr)

The screen displays:

```
4:*_
```

If you now enter:

```
4:*REM NEWDISK was created with EDLIN(cr)
5:*^Z(cr)
```

and type the L command to list the file, the result is:

```
1:REM This is a file to check formatted disks
2:REM This file is a batch file
3:REM This batch file is called NEWDISK
4:REM NEWDISK was created with EDLIN
5:*PAUSE Insert formatted disk in drive B
6:DIR B:
7:CHKDSK B:
```


List Lines (L)

[line][,line]L(cr)

"Line" can be any of the parameters for a line described in Table 4-1.

L lists the specified range of lines, as follows:

- With no parameters, L lists the 23 lines centered around the current line (marked by an *). The 23 lines are the 11 lines preceding the current line, the current line, and the 11 lines following the current line.
- line,line or line,# lists the specified range of lines.

- ,line lists the range of lines preceding the specified line.
- line, lists a range of lines following the specified line.

Assume the following file exists and is ready to edit:

```
1:REM This is a file to check formatted disks
2:REM It is named NEWDISK.BAT
3:PAUSE Insert formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

To list the entire file, enter:

```
L(cr)
```


All five lines are displayed, because L without parameters displays up to 23 lines of text.

To list a range of lines preceding line 3, enter:

```
,3L(cr)
```

The result is:

```
1:REM This is a file to check formatted disks
2:REM It is named NEWDISK.BAT
3:PAUSE Insert formatted disk in drive B
```

To list a range of lines following line 3, enter:

```
3,L(cr)
```

The result is:

```
3:PAUSE Insert formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

Quit Edit (Q)

Q(cr)

Q quits the editing session without saving the editing changes, and exits to MS-DOS. The Q command is a quick way to exit an editing session. When the Q command is given, EDLIN displays the message:

```
Abort edit (Y/N)?
```

Press Y to quit the editing session; press N (or any other key except CTRL-C) to continue the editing session.

Replace String (R)

[line][,line][?]Rstring1[^Z]string2(cr)

R replaces string1 with string2, in the range of lines specified by "line".

"Line" can be any of the parameters for a line described in Table 4-1. If no line is indicated, R operates on the line after the current line (marked by an *) through the end of the file.

- line is the specified line to the end of the file.
- line, line is the first specified line through the second specified line.
- ,line is the line after the current line through the specified line.
- .,line is the current line through the specified line.
- line,. is the range of lines from the specified line through the current line.

? prompts you to OK each string replacement or deletion. When EDLIN encounters a matching string of text, it displays the line and the prompt "O.K.?". If you press Y or Return, EDLIN makes the replacement or deletion and searches for the next matching string. If you press any key other than Y or Return, EDLIN leaves the string as is and searches for the next matching string.

string1 is the string of characters to be replaced. string2 is the string of characters to replace string1. CTRL-Z separates string1 from string2.

If you specify only one string of characters, R deletes all occurrences of the string within the specified range of lines.

In this example, the following file exists and is ready for editing:

```
1:REM This is a file to check a formatted disk
2:REM It is named NEWDISK.BAT
3:PAUSE Insert a formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

To replace all occurrences of "a" with "one" in lines 1 through 3, enter:

```
1,3Ra^Zone(cr)
```

The result is:

```
1:*REM This is one file to check one formonetted
  disk
2:It is nonemed NEWDISK.BAT
3:PAUSE Insert one formonetted disk in drive B
```

Some of these replacements produce nonsense words. To avoid unwanted substitutions, add the ? parameter:

```
1,3?Ra^Zone(cr)
```

Now each specified line is displayed, and you can confirm the changes. For example:

```
     1:*REM This is one file to check a formatted
       disk
O.K.?y
     1:*REM This is one file to check one
       formatted disk
O.K.?y
     1:*REM This is one file to check one
       formonetted disk
O.K.?n
     2:REM It is nonemed NEWDISK.BAT
O.K.?n
     3:PAUSE Insert one formatted disk in drive B
O.K.?y
```


After making the replacements you want, enter an L command to list the edited file:

```
*L(cr)
1:REM This is one file to check one formatted disk
2:REM It is named NEWDISK.BAT
3:*PAUSE Insert one formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

Search for a String (S)

[line][,line][?]Sstring(cr)

S searches for the string in the range of lines specified by line. It displays the first line containing the string. This line becomes the current line if the ? parameter is not specified.

"Line" can be any of the parameters for a line described in Table 4-1. The syntax for "line" is given in the Replace command description. If you do not specify a line, S operates on the line after the current line through the end of the file.

The ? prompts you to OK the displayed line as the current line. When EDLIN encounters a matching string of text, it displays the line and the prompt "O.K.?". If you press Y or Return, the displayed line becomes the current line. If you press any key other than Y or Return, EDLIN searches for the next matching string.

Assume the following file exists and is ready for editing:

```
1:REM This is a file to check a formatted disk
2:REM It is named NEWDISK.BAT
3:PAUSE Insert a formatted disk in drive B
4:DIR B:
5:CHKDSK B:
```

To search for the first "in" within lines 1 through 3, enter:

```
1,3Sin(cr)
```

The result is:

```
3:PAUSE Insert a formatted disk in drive B
```

To find "DIR" past line 3, specify:

3SDIR(cr)

The search starts at line 3 and continues to the end of the file. EDLIN displays the line containing the first "DIR":


```
4:DIR B:
```

To search the file for the first occurrence of "B", at the * prompt enter:

?SB(cr)

The result is:

```
     2:REM It is named NEWDISK.BAT
O.K.?_
```

If you respond Y, the search ends and you return to the * prompt. To continue the search, respond N.

Write Lines (W)

[n]W(cr)

W writes lines from memory to disk and makes room in memory for more lines to be appended (with the Append command) from disk to memory.

n represents the number of lines to be written.

If no parameter is given, lines are written to disk until memory is only 1/4 full. If a number is given, the specified number of lines is written to disk. As lines are written, subsequent lines are renumbered beginning with 1.

An A command following the W command appends lines from disk to any line remaining in memory.